

This Page Is Inserted by IFW Operations
and is not a part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

**As rescanning documents *will not* correct images,
please do not report the images to the
Image Problem Mailbox.**

THIS PAGE BLANK (USPTO)

E5887

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-29956

(P2000-29956A)

(43) 公開日 平成12年1月28日 (2000.1.28)

(51) Int.Cl.⁷

G 0 6 F 19/00

識別記号

F I

G 0 6 F 15/22

テーマコード* (参考)

N

審査請求 未請求 請求項の数5 O L (全 10 頁)

(21) 出願番号 特願平10-192548
(22) 出願日 平成10年7月8日 (1998.7.8)

(71) 出願人 000005223
富士通株式会社
神奈川県川崎市中原区上小田中4丁目1番1号
(72) 発明者 原 裕貴
神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内
(72) 発明者 金谷 延幸
神奈川県川崎市中原区上小田中4丁目1番1号 富士通株式会社内
(74) 代理人 100087848
弁理士 小笠原 吉義 (外2名)

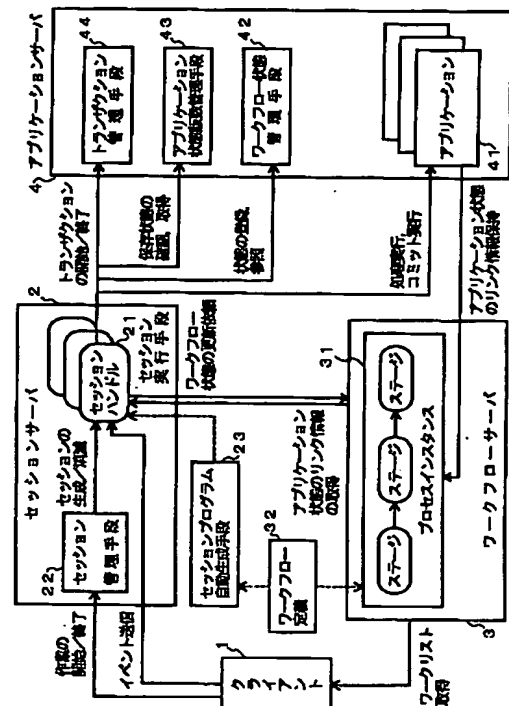
最終頁に続く

(54) 【発明の名称】 ワークフローアプリケーション実行装置およびワークフローアプリケーション構築方法

(57) 【要約】

【課題】 ワークフローアプリケーションを構築し実行する装置に関し、ワークフロー実行中に、状態を変更するようなアプリケーションを用いることができ、トランザクション処理を行うことができ、ワークフロー状態とアプリケーションの状態との整合を維持できるようにすることを目的とする。

【解決手段】 ワークフロー定義32中に、アプリケーション41の状態保持、トランザクション処理、コミット等の記述を可能にし、これをもとにセッションプログラムのスケルトンを自動生成する。ワークフローサーバ3は、各ステージごとにアプリケーション状態のリンク情報を保持する。セッション管理手段22は、各ステージごとにセッションを生成する。セッションでは、ワークフローサーバ3からアプリケーション状態のリンク情報を取得して、クライアント1とアプリケーション41との対話処理を遂行する。



【特許請求の範囲】

【請求項1】 一連の業務の遂行に必要な複数の処理に関する各々のステージの実行順序を定めるワークフローの定義にもとづき、外部の業務アプリケーションを用いて各ステージごとに対応する処理を行いながら全体として前記業務を遂行するワークフローアプリケーション実行装置であって、前記業務アプリケーションへのリンク情報を保持し、ワークフローを制御する手段と、前記ワークフローの各ステージにおいてクライアントと業務アプリケーションとの対話処理を遂行するためのセッションを、作業開始時に生成し、作業終了時に消滅させるセッション管理手段と、前記セッション管理手段により生成されたセッションからの処理要求により、前記業務アプリケーションを実行する手段とを備えることを特徴とするワークフローアプリケーション実行装置。

【請求項2】 請求項1記載のワークフローアプリケーション実行装置において、前記セッション管理手段は、前記ワークフローにおける次のステージの担当者が同じクライアントであるとき、セッションを切り替えずにクライアントの作業用の画面のみを切り替えて処理を実行させることを特徴とするワークフローアプリケーション実行装置。

【請求項3】 一連の業務の遂行に必要な複数の処理に関する各々のステージの実行順序を定めるワークフローの定義にもとづき、外部の業務アプリケーションを用いて各ステージごとに対応する処理を行いながら全体として前記業務を遂行するワークフローアプリケーション実行装置を構築するための方法であって、ワークフロープロセスの定義により、前記ワークフローの各ステージにおいてクライアントと業務アプリケーションとの対話処理を遂行するためのセッションを実現するプログラムの一部を自動生成することを特徴とするワークフローアプリケーション構築方法。

【請求項4】 請求項3記載のワークフローアプリケーション構築方法において、ワークフロープロセスの定義中に、トランザクションの開始およびコミットを明示的または暗黙的に記述させ、それをもとにワークフローの実行に同期してトランザクション処理を実行するための前記セッションを実現するプログラムを自動生成することを特徴とするワークフローアプリケーション構築方法。

【請求項5】 一連の業務の遂行に必要な複数の処理に関する各々のステージの実行順序を定めるワークフローの定義にもとづき、外部の業務アプリケーションを用いて各ステージごとに対応する処理を行いながら全体として前記業務を遂行するワークフローアプリケーション実行装置を構築するための方法であって、ワークフロープロセスの定義中のステージに対してアプリケーションの状態を保持することを記述することを可能とし、ワークフローの状態が差戻し等によって前の状態に戻ったと

き、アプリケーションの状態もその状態に戻すことを特徴とするワークフローアプリケーション構築方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、ワークフローシステムに関し、特にワークフローシステムがデータベース処理等の既存アプリケーションや、ビジネスオブジェクト等のバックエンドアプリケーションと協調して、複数のクライアントにまたがって作業が実行されるような分散アプリケーションの構築および実行ができるようにしたワークフローアプリケーション実行装置およびワークフローアプリケーション構築方法に関する。

【0002】

【従来の技術】たとえば、一般の企業において、稟議書や伝票等は起票者から上司へ、さらにその上司へというように、必要に応じてあらかじめ設定されたルートで回覧され、承認、決裁される。このように、ある業務に関する作業の一連の流れ、作業内容によって定型化された一連の業務の流れを指してワークフローと呼ぶ。ワークフローシステムは、グループウェアの代表的な機能の一つであり、たとえば上で述べたような文書の流れを電子メッセージの流れに代え、必要なルートに従ったメッセージの配信およびその管理などを行なう。

【0003】通常、ワークフローシステムのサーバは、状態を保持したり、外部のアプリケーションを実行したりするための簡単な処理手段しか持っていない。このため、従来のワークフローシステムでは、ユーザとの対話によって入力された情報をワークフローシステムの内部に蓄え、プロセスの終了時にデータベースに書き込むような単純な作業しか行うことができなかった。また、画面の定義や外部アプリケーションとのやりとりを行う実行プログラムや状態の管理手段を、クライアントおよびサーバの両方に実装する必要があった。

【0004】また、ワークフローシステム自体は、通常、トランザクション機能を持たないので、トランザクション処理は別途用意して行う必要があった。

【0005】

【発明が解決しようとする課題】以上のような従来のワークフローシステムでは、ワークフローが変更された際に、サーバとクライアントの両方のプログラムを整合的に書き換える必要があり、ワークフローの開発上の大きな問題となっていた。

【0006】また、従来の技術では、ワークフロー実行中に、複雑な処理を伴うアプリケーションを呼び出して状態を書き換えたり、既存の業務アプリケーションをワークフローシステムによって制御し、複数のクライアントによる操作を経て一つのプロセスを終了したりするようなワークフローアプリケーションとして、システムを運用することは困難であった。

【0007】また、クライアントやサーバ上でデータを

保持すると、トランザクション処理が行えないという問題があった。さらに、外部プログラムでトランザクション処理を行った場合には、サーバのダウンなどの際に、サーバで保持しているワークフロー状態との不整合が発生してしまうという問題があった。この場合に、ワークフロー状態も含めてトランザクション処理に入れてしまうと、ワークフローの処理が遅くなってしまうという問題もあった。

【0008】本発明の目的は、ワークフロー実行中に、既存アプリケーションをワークフローシステムによって制御し、複数のクライアントでの操作を経て一つのプロセスを終了するようなワークフローアプリケーションとして利用でき、また、ワークフロー処理を遅延させずにトランザクション処理を行え、ワークフローの各ステージとアプリケーションとの状態の不整合を起こさないようなワークフローアプリケーションを構築し、かつ実行する装置を実現することである。

【0009】

【課題を解決するための手段】図1は、本発明の構成例を示す図である。本発明は、クライアント1からのイベント（処理要求）を受け取り、セッションサーバ2に、ワークフローや業務プロセスの処理要求を発行するセッション実行手段21（以下、セッションハンドルという）を持ち、ワークフローサーバ3内のワークフロー定義32にもとづくプロセスインスタンス31やワークアイテムが外部アプリケーションへのリンク情報を保持する機構を持つように構成したことを特徴とする。

【0010】ワークフローサーバ3とクライアント1は、フレームワークによって定められた基本的な処理のみを実行する。セッション管理手段22は、セッションの生成と消滅を管理し、ワークフローサーバ3の持つプロセスインスタンス31の一つのステージに対して一つのセッションハンドル21を生成する。

【0011】セッションハンドル21は、ワークフローの一つのステージの間のみ存在し、そのステージに対応するセッションプログラムに記述された処理を行い、処理が終了すると消滅し、ステージを進めるイベントを発行する。セッションハンドル21は、クライアント1に対する画面の一時的な状態を保持するが、アプリケーションやワークフローの状態を保持することはしない。

【0012】また、セッションハンドル21のセッションプログラム中に、アプリケーション41のトランザクションの開始、コミット等を記述することができるようになっている。これにより、トランザクションの開始／終了、アプリケーションの保存状態の確認、取得、ワークフロー状態の登録、参照、処理の実行、コミット実行等の指示を行うことができる。

【0013】セッションプログラム自動生成手段23は、トランザクションの開始、コミット、アプリケーション状態の版数管理等を記述したワークフロー定義32

から、セッションハンドル21のスケルトン（セッションプログラムのひな型）を自動生成する。

【0014】ワークフローサーバ3のプロセスインスタンス31は、アプリケーションの状態を持つことはせず、アプリケーションの状態（ビジネスオブジェクト等）に対するリンク情報（キー等）のみを持ち、各ステージは、この状態へのリンク情報（キー等）を承継することにより、いつでもアプリケーションの状態を取得することができる。たとえば伝票処理の場合には、このリンク情報のキーとして、伝票番号のようなものが用いられる。

【0015】アプリケーションサーバ4のワークフロー状態管理手段42は、現在のワークフロー状態情報を保持し、アプリケーション状態版数管理手段43は、アプリケーション41の状態情報を保持する。これらの情報により、なんらかの原因でワークフロー状態とアプリケーションの状態とが不整合になったことを確認することができ、たとえば、ワークフローのステージをロールバックさせることで、これらの状態の整合性が保たれる。

【0016】トランザクション管理手段44は、ワークフローの実行に同期したトランザクションの管理を行う。このトランザクションの管理では、一つのステージ内に閉じたショートトランザクションと、複数のステージにまたがったトランザクションとショートトランザクションとのネステッドトランザクションの管理を行うこともできる。これにより、アプリケーションサーバ4のトランザクション管理機能を用いて、ワークフローシステム全体のトランザクション処理の実現が可能となる。

【0017】本発明は、以下のように作用する。セッションハンドル21は、クライアント1からのワークフロー定義32に定義されたプロセスインスタンス31の一つのステージを実行するという要求に対して生成される。

【0018】セッションハンドル21は、クライアント1のGUIイベント（処理要求）を受け取り、適切なアプリケーション41の処理を実行する。これは通常のアプリケーション41でコントロールと呼ばれるモジュール（処理制御手段）が行うのと同じ処理である。通常のコントロールと違うのは、クライアント1の特定のイベントに対してはワークフローサーバ3にワークフロー処理を要求することである。

【0019】ワークフローサーバ3によって、このステージの終了が判断された際には、セッションハンドル21は、クライアント1側に画面消去などの終了処理を行った後、セッションハンドル21自体も消滅する。

【0020】このセッションハンドル21自体は、一つのステージの間のみライフサイクルを持つので、ステージをまたがるような状態を保持することができない。したがって、アプリケーション41の状態は、アプリケーション状態版数管理手段43によって、アプリケーシ

ョンサーバ4内に保持され、次のステージにおいて生成されるセッションハンドル21がアプリケーション41を実行するために必要なリンク情報（ビジネスオブジェクトのID、データベースのテーブルIDなど）は、ワークフローサーバ3内のプロセスインスタンス31やプロセスに登録して管理する。

【0021】ワークフローサーバ3に登録するのは、状態に関するリンク情報のみであり、各セッションの処理の内容は、セッションハンドル21に記述される。なお、セッションハンドル21が動作するのはクライアント1上でも構わないし、プロセスインスタンス31と同一プロセスで動作するようにしてもよい。

【0022】

【発明の実施の形態】以下、本発明の実施の形態を説明する。図2は、本発明の基本的な処理フローチャートである。

【0023】図2において、クライアント1は、ワークフローサーバ3よりワークリストを取得し（ステップS1）、ワークリストより作業を選択する（ステップS2）。これにより、作業開始がセッションサーバ2に送られ、セッションハンドル21が生成されて、セッションが開始する（ステップS3）。セッションは、ワークフローサーバ3から対応するプロセスに関連するアプリケーション41の状態に対するリンク情報を取得し（ステップS4）、また、クライアント1に画面を表示する（ステップS5）。

【0024】クライアント1からセッションに対してイベントが通知されると（ステップS6）、セッションはイベントに対応した処理を実行する（ステップS7～S8）。たとえば、セッションは、クライアント1からのイベントに応じてアプリケーション41に対して処理を要求し（ステップS9）、またはトランザクションの開始／終了、コミット処理を要求する（ステップS10）。あるいは、ステージ終了のイベントに対しては、ワークフローサーバ3に対してそのステージの終了処理を依頼し（ステップS11）、クライアント1に対してもステージの終了処理を依頼し（ステップS12）、セッション自身は消滅する（ステップS13）。

【0025】通常ステージが移動する際には、セッションハンドル21（セッション）は消滅し、次のステージの作業が担当者のワークリストに載る。しかし、同じ担当者が連続するステージを処理する場合には、いちいちワークリストから作業を選択することなく途切れずに処理を続行できれば作業効率を上げることができる。そこで、このような場合には、古いセッションを消去するとともに新しいセッションを生成し、クライアント1に対して新しいセッションに対応する画面を送ることによって、クライアント1が連続的に作業を行えるようにする。

【0026】図3は、同一担当者によるステージ実行の

処理フローチャートである。セッション終了のイベントに対して、セッションは、ワークフローサーバ3に対してステージの終了処理を依頼し（ステップS21）、クライアント1に対してステージの終了処理を依頼し（ステップS22）、セッション自身は消滅する（ステップS23）。このとき、ワークフローサーバ3は、次のステージの担当者が同一かどうかを判断し（ステップS24）、同一でない場合には、ワークフローサーバ3は、次のステージの作業を担当者のワークリストに登録する（ステップS25）。

【0027】次のステージの担当者が同一の場合には、ワークフローサーバ3は、セッション管理手段22に対して次のセッションの生成、開始を要求する（ステップS26）。生成されたセッションは、ワークフローサーバ3から対応するプロセスに関連するアプリケーション41の状態に対するリンク情報を取得し（ステップS27）、同時に、クライアント1に画面を表示する（ステップS28）。

【0028】また、アプリケーションサーバ4側に用意されたアプリケーション状態版数管理手段43は、アプリケーション41の状態版数を管理する。セッションは、必要なステージに対応するアプリケーション41の状態版数の保持を要求する。たとえば、ステージの差し戻し等により、再び同じステージの処理を行うことになった場合に、アプリケーション41の状態版数情報が保存されていれば、その状態に戻すことができ、状態の整合性が維持できる。

【0029】図4は、アプリケーション状態版数保持の処理フローチャートである。図4において、セッションによってステージの処理が開始されると（ステップS31）、セッションは、該当ステージに対応する状態が存在するかどうかをアプリケーション状態版数管理手段43に対して聞く（ステップS32）。

【0030】該当ステージに対応する状態が存在する場合には、セッションは、アプリケーション状態版数管理手段43からアプリケーション41の状態を取得する（ステップS33）。状態が存在しない場合には、セッションは、ワークフローサーバ3から対応するプロセスに関連するアプリケーション41の状態に対するリンク情報を取得する（ステップS34）。

【0031】ステージの終了が指示されると（ステップS35）、そのステージがアプリケーション41の状態を保存するように指定されているステージかどうかを判断し（ステップS36）、状態を保存するように指定されているステージである場合には、アプリケーション状態版数管理手段43に対して、状態を保存するように要求する（ステップS37）。

【0032】また、アプリケーションサーバ4側に用意されたワークフロー状態管理手段42は、ワークフローの状態を保持する。セッションは、トランザクション管

理手段44により、各ステージの処理におけるトランザクション処理を行うようにする。セッションは、このトランザクションが正常に終了したことを確認した上でワークフローの状態を次に進め、ワークフローとアプリケーション41の状態が整合するように保つ。

【0033】図5は、同期処理による状態整合維持の処理フローチャートである。図5において、セッションを開始し(ステップS41)、トランザクションを開始する(ステップS42)。続いて、セッションを実行し、アプリケーション処理を行う(ステップS43)。ステージの終了が指示され(ステップS44)、アプリケーションサーバ4にワークフロー状態を登録し(ステップS45)、トランザクションを終了する(ステップS46)。トランザクションが成功したかどうかを判断し(ステップS47)、トランザクションが成功した場合にはワークフローの状態を変更する(ステップS48)。

【0034】また、トランザクション処理を待たずにワークフローの状態を先に進めることも可能である。この場合には、ワークフローサーバ3は、各ステージの動作を開始する際に、アプリケーション側の状態を確認した上で、そのステージの動作を再開する。状態が不整合の場合(ワークフロー状態は進んでいるのに、アプリケーション側でトランザクションが失敗して状態が変わっていない場合、または、アプリケーション側のトランザクションが成功しているのに、ワークフローのステージが進んでいない場合)には、ワークフロー側は、アプリケーション41の状態に合わせるように、ステージを移行する処理を行い、整合性を維持する。

【0035】図6は、非同期処理による状態整合維持の処理フローチャートである。図6において、セッションを開始し(ステップS51)、アプリケーション41の状態とワークフロー状態との整合がとれているかどうかを判断し(ステップS52)、状態の整合がとれていない場合には、セッションを終了し(ステップS53)、ワークフロー状態をアプリケーション41の状態と整合するようなステージに移行する(ステップS54)。その後、ステップS51へ戻り、新たなセッションを開始する。

【0036】状態の整合がとれている場合には、トランザクションを開始し(ステップS55)、セッションを実行し、アプリケーション処理を行う(ステップS56)。その後、ステージの終了が指示されると(ステップS57)、ワークフローの状態を変更するとともに(ステップS58)、アプリケーションサーバ4にワークフロー状態を登録し(ステップS59)、トランザクションを終了する(ステップS60)。なお、ステップS58またはステップS60の処理は失敗することもある。

【0037】また、ワークフロー定義中にトランザクシ

ョンの開始/終了、コミット、状態保持を行うなどの情報を記述できるようにして、これらの情報にもとづいて、セッションプログラムの一部(ひな型)を自動生成する。

【0038】図7は、ワークフロー定義を用いてセッションプログラムを自動生成する処理を説明する図である。セッションプログラム自動生成手段23は、ワークフロー定義32中に、トランザクションの開始、状態保持、コミット、トランザクション終了などの記述があると、それをもとに各ステージにおいて実行するセッションプログラムを自動生成する。たとえば、ワークフロー定義32中のステージに対してトランザクション開始、状態保持を定義した場合には、「セッション開始、トランザクション開始、状態保持、…、アプリケーションの処理実行、…、セッション終了」のようなセッションプログラムのスケルトンを自動的に生成する。

【0039】

【実施例】図8および図9を用いて、ある研究所における物品等購入業務を例にして、ワークフローアプリケーションの実行制御を説明する。

【0040】物品等購入業務では、クライアント1として起案者、上司、購買担当者が物品等購入ワークフローに関係して、研究所予算管理アプリケーションおよび予算利用申請伝票処理アプリケーションを使用して各クライアントが個別に処理を行いながら、業務を遂行していく。

【0041】物品等購入ワークフローは、図8に示すようなワークフロー定義にもとづいて進行する。ワークフロー定義には、どのような順序でステージを処理するのか、各ステージでは誰が処理するのか、どのようなときに処理をするのか等を定義しておく。ステージは、そのワークフローで行われる一連の処理動作のうち、一つのまとまった処理の単位である。一つのステージは一つのセッションで実行される。

【0042】図8の物品等購入ワークフローは、予算使用申請、上司承認、購買見積もり、実績更新、成功処理、失敗処理の各ステージからなる。各ステージの関係は、まず、予算使用申請ステージで起案者により予算使用申請セッションが処理され、次に上司承認ステージで上司により上司承認セッションが処理される。上司承認セッションで「承認」がされた場合には購買見積もりステージに進む。購買見積もりステージで購買担当者により購買見積もりセッションが処理され、購買見積もりが予算より少ない場合には、実績更新ステージに進む。実績更新ステージで実績更新セッションが処理され、実績更新が成功した場合には、起案者に対する成功処理ステージへ進む。

【0043】なお、上司承認ステージで「承認」されなかった場合、購買見積もりステージで購買見積もりが予算より多い場合、実績更新ステージで失敗した場合に

は、起案者に対する失敗処理ステージに進む。

【0044】物品等購入ワークフローの各処理（セッション）の詳細を、図9を用いて説明する。まず、起案者は、物品等購入業務をすすめるため購買ワークフローシステムにアクセスし、「予算使用申請」セッションを開始する。「予算使用申請」セッションでは、研究所予算管理アプリケーションを用いて、予算オブジェクトを検索し、実績・予算を取得してクライアント1の端末に実績・予算を表示し、予算の申請を入力させ、予算を減額する。さらに、予算利用申請伝票処理アプリケーションを用いて、予算申請の伝票を生成し、購入内容、実績・予算登録を行う。セッションが終了すると、別のクライアントである上司のワークリストに「上司承認」の作業が登録され、次のステージに進む。

【0045】上司が購買ワークフローシステムにアクセスし、ワークリストから「上司承認」の作業を選択すると、「上司承認」セッションが開始する。「上司承認」セッションでは、予算利用申請伝票処理アプリケーションを用いて、起案者が入力した内容、実績・予算を取得して表示し、承認結果を入力する。ここで、「承認」が入力されてセッションが終了すると、購買担当者（購買）のワークリストに「購買見積もり」の作業が登録され、次のステージに進む。

【0046】購買担当者が購買ワークフローシステムにアクセスし、ワークリストから「購買見積もり」の作業を選択すると、「購買見積もり」セッションが開始する。「購買見積もり」セッションでは、予算利用申請伝票処理アプリケーションから上司が承認した内容、実績・予算を取得して、クライアント1に表示し、見積もり結果を入力し、見積もり額を予算利用申請伝票処理アプリケーションに登録する。見積もり額が予算を超えない場合、「実績更新」セッション起動のイベントが発行される。

【0047】次のステージの「実績更新」セッションでは、研究所予算管理アプリケーションの実績・予算を修正する。実績更新が成功してセッションを終了すると、起案者のワークリストに「成功処理」が登録される。

【0048】次に起案者が購買ワークフローシステムにアクセスし、「成功処理」セッションが開始すると、起案者の端末に「承認」の結果が表示され、予算利用申請の伝票が削除され、セッションが終了する。

【0049】なお、「上司承認」セッションにおいて、「否認」が入力されて「上司承認」セッションが終了する場合には、起案者のワークリストに「失敗処理」が登録される。これより、起案者が次にアクセスしたときに開始される「失敗処理」セッションにおいて、「否認」結果が表示され、予算利用申請の伝票が削除される。

「購買見積もり」セッションにおいて、見積もり額が予算を超過した場合または「実績更新」セッションにおける実績更新が失敗した場合も同様に、「失敗処理」セッ

ションによる結果表示および伝票削除の処理が行われる。

【0050】たとえば、以上の各ステージにおける個々の処理をショートトランザクションとして扱うとともに、それらの全体を、トランザクションのネスティングによる上位トランザクションとして扱うようにワークフロー定義32を記述し、セッションプログラムを自動生成することによって、ワークフローシステムがトランザクション管理機能を利用し、ワークフローとアプリケーションとの連携および整合性を簡易にとることができるようになる。

【0051】

【発明の効果】以上説明したように、本発明によれば、従来のワークフローシステムのように、ワークフロー定義をクライアント側とワークフローサーバ側に持つ必要がなく、ワークフロー定義の修正の処理負担が軽くなるという効果を奏する。

【0052】また、連続するステージの担当者が同一クライアントの場合には、セッションを切り替えずにクライアント側の画面のみを切り替えて処理を実行することができ、操作性を向上させるという効果を奏する。

【0053】また、ワークフロー実行中に、状態の書き換えを伴うような複雑なアプリケーションや既存の業務アプリケーションを、ワークフローアプリケーションとして使用することが可能となる。

【0054】また、ワークフローシステムにおいても、処理速度を落とさずに簡易にトランザクション処理が可能となり、さらにアプリケーションの状態とワークフロー状態とが不整合になった場合にも、状態を整合させることが可能となる。

【0055】以上の説明に関して、さらに以下の項を開示する。

(1) 一連の業務の遂行に必要な複数の処理に関する各々のステージの実行順序を定めるワークフローの定義にもとづき、外部の業務アプリケーションを用いて各ステージごとに対応する処理を行いながら全体として前記業務を遂行するワークフローアプリケーション実行装置であって、前記業務アプリケーションへのリンク情報を保持し、ワークフローを制御する手段と、前記ワークフローの各ステージにおいてクライアントと業務アプリケーションとの対話処理を遂行するためのセッションを、作業開始時に生成し、作業終了時に消滅させるセッション管理手段と、前記セッション管理手段により生成されたセッションからの処理要求により、前記業務アプリケーションを実行する手段とを備えるワークフローアプリケーション実行装置において、前記セッションの中で、アプリケーション処理のトランザクションの開始、コミットを実行することによって、ワークフローの実行に同期してトランザクション処理を実行するように構成する。

【0056】(2) また、一連の業務の遂行に必要な複

数の処理に関する各々のステージの実行順序を定めるワークフローの定義にもとづき、外部の業務アプリケーションを用いて各ステージごとに対応する処理を行いながら全体として前記業務を遂行するワークフローアプリケーション実行装置であって、前記業務アプリケーションへのリンク情報を保持し、ワークフローを制御する手段と、前記ワークフローの各ステージにおいてクライアントと業務アプリケーションとの対話処理を遂行するためのセッションを、作業開始時に生成し、作業終了時に消滅させるセッション管理手段と、前記セッション管理手段により生成されたセッションからの処理要求により、前記業務アプリケーションを実行する手段とを備えるワークフローアプリケーション実行装置において、前記ワークフローと前記アプリケーションの状態が非整合になったとき、前記ワークフローの状態を移行させ、状態を同期させる手段を備える。

【図面の簡単な説明】

【図1】本発明の構成例を示す図である。

【図2】本発明の基本的な処理フローチャートである。

【図3】同一担当者によるステージ実行の処理フローチャートである。

【図4】アプリケーション状態版数保持の処理フローチャートである。

【図5】同期処理による状態整合維持の処理フローチャ

ートである。

【図6】非同期処理による状態整合維持の処理フローチャートである。

【図7】ワークフロー定義を用いてセッションプログラムを自動生成する処理を説明する図である。

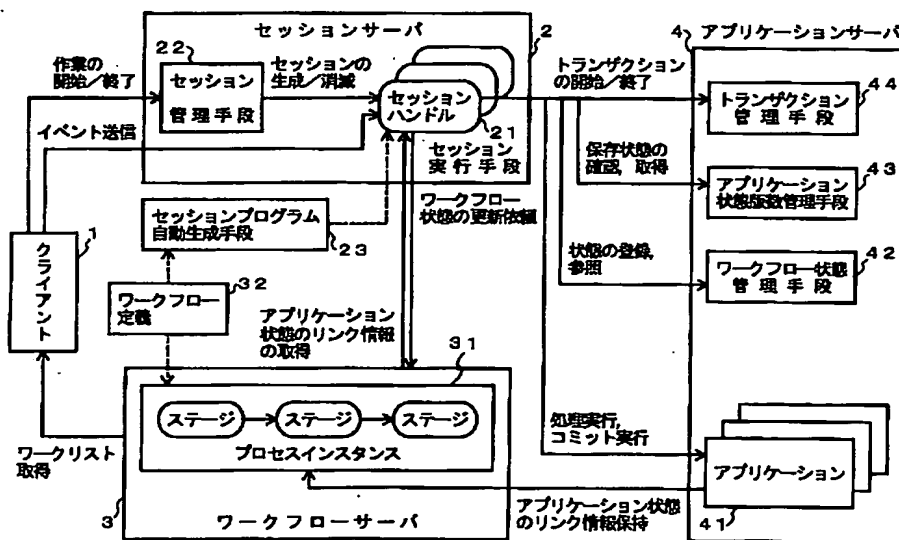
【図8】ワークフロー定義（プロセスインスタンス）の例を示す図である。

【図9】ワークフロー制御の実施例を説明する図である。

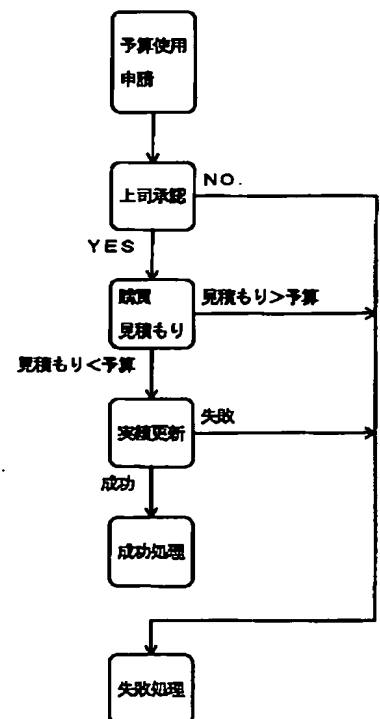
【符号の説明】

- 1 クライアント
- 2 セッションサーバ
 - 21 セッション実行手段（セッションハンドル）
 - 22 セッション管理手段
 - 23 セッションプログラム自動生成手段
- 3 ワークフローサーバ
 - 31 プロセスインスタンス
 - 32 ワークフロー定義
- 4 アプリケーションサーバ
 - 41 アプリケーション
 - 42 ワークフロー状態管理手段
 - 43 アプリケーション状態版数管理手段
 - 44 トランザクション管理手段

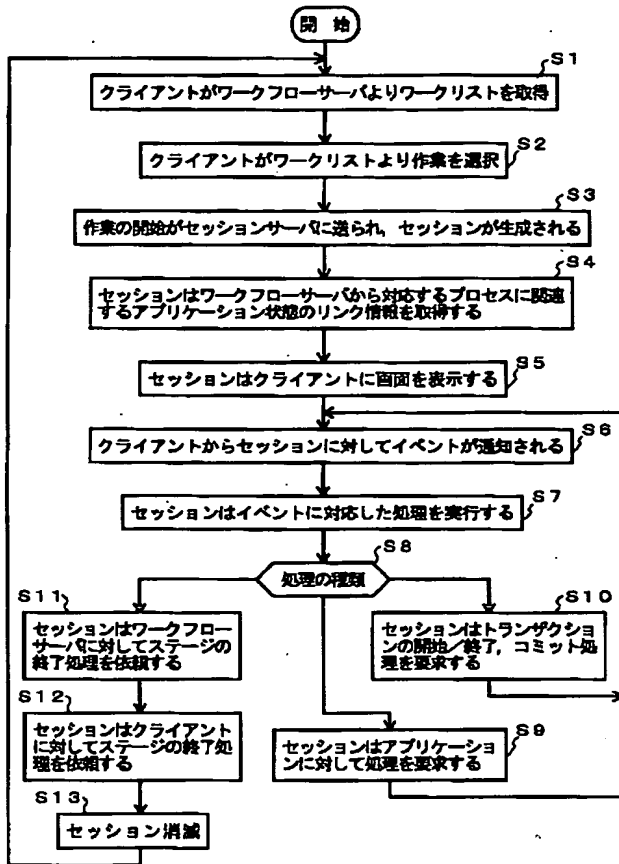
【図1】



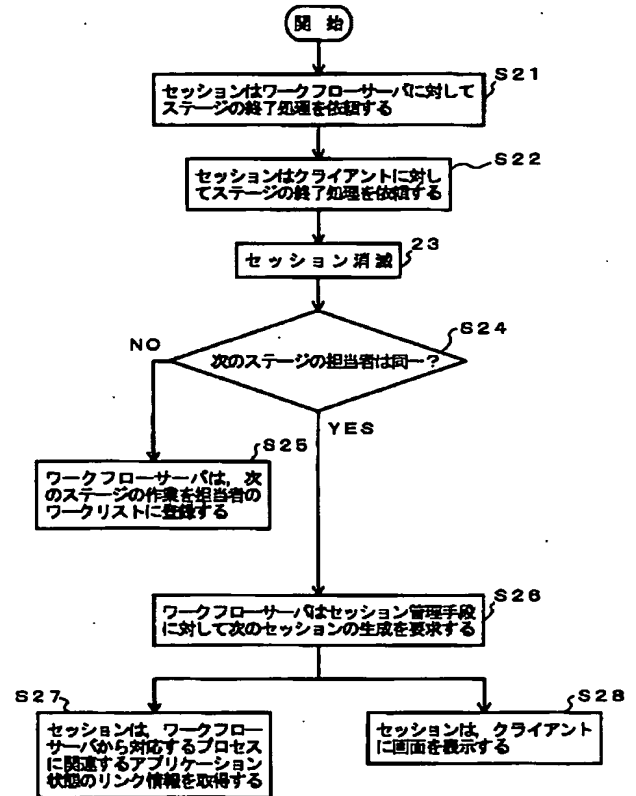
【図8】



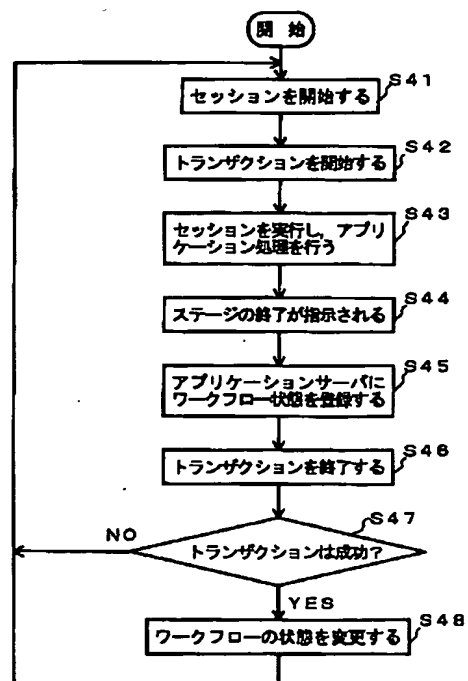
【図2】



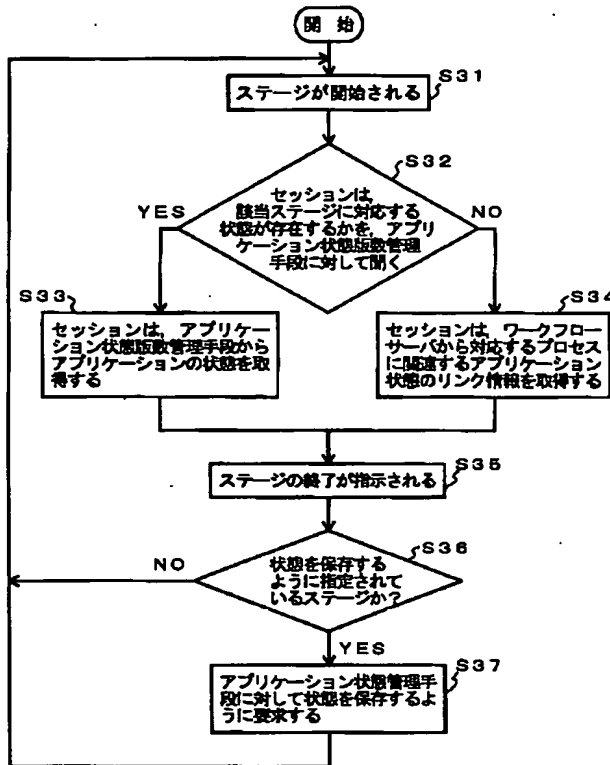
【図3】



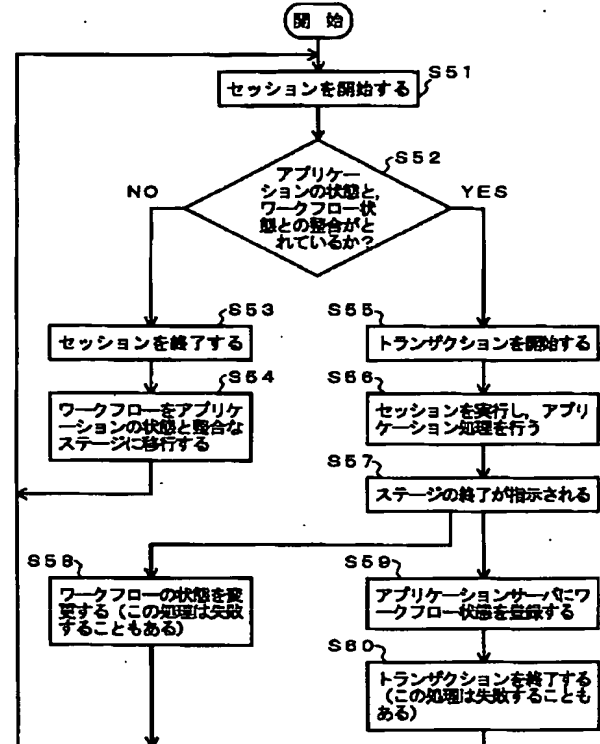
【図5】



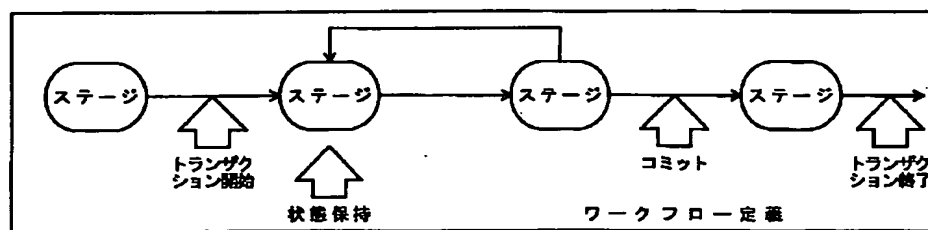
【図4】



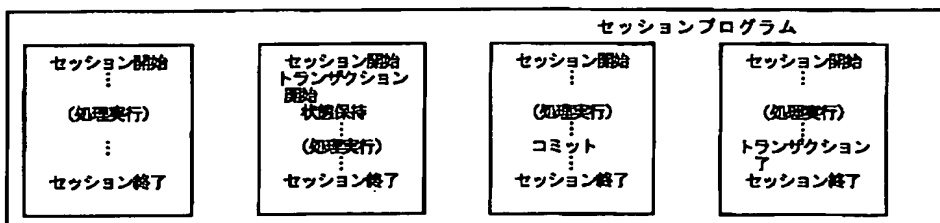
【図6】



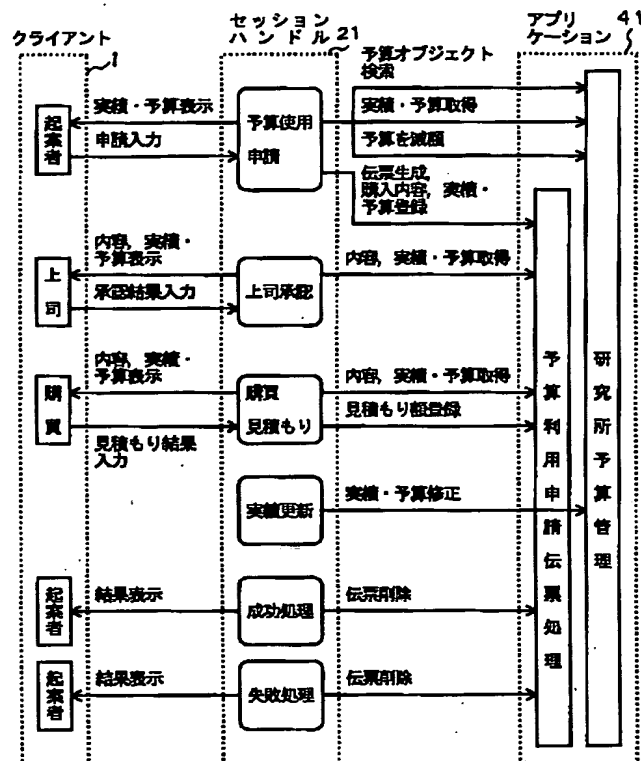
【図7】



自動生成



【図9】



フロントページの続き

(72)発明者 松塚 貴英
神奈川県川崎市中原区上小田中4丁目1番
1号 富士通株式会社内

THIS PAGE BLANK (USPTO)